International Journal of Advanced Culture Technology Vol.9 No.4 288-294 (2021)
DOI https://doi.org/10.17703/IJACT.2021.9.4.288

IJACT 21-12-34

A study on Countermeasures by Detecting Trojan-type Downloader/Dropper Malicious Code

Hee Wan Kim

Prof., Division of Computer Science & Engineering, Sahmyook Univ., Korea
hwkim@syu.ac.kr

Abstract

There are various ways to be infected with malicious code due to the increase in Internet use, such as the web, affiliate programs, P2P, illegal software, DNS alteration of routers, word processor vulnerabilities, spam mail, and storage media. In addition, malicious codes are produced more easily than before through automatic generation programs due to evasion technology according to the advancement of production technology. In the past, the propagation speed of malicious code was slow, the infection route was limited, and the propagation technology had a simple structure, so there was enough time to study countermeasures. However, current malicious codes have become very intelligent by absorbing technologies such as concealment technology and self-transformation, causing problems such as distributed denial of service attacks (DDoS), spam sending and personal information theft. The existing malware detection technique, which is a signature detection technique, cannot respond when it encounters a malicious code whose attack pattern has been changed or a new type of malicious code. In addition, it is difficult to perform static analysis on malicious code to which code obfuscation, encryption, and packing techniques are applied to make malicious code analysis difficult. Therefore, in this paper, a method to detect malicious code through dynamic analysis and static analysis using Trojan-type Downloader/Dropper malicious code was showed, and suggested to malicious code detection and countermeasures.

Keywords: *Malicious code, Dynamic analysis, Static analysis, Malicious code detection, Countermeasures*

1. INTRODUCTION

As the spread of personal PCs increases and the use of the Internet increases, the routes of infection by malicious code have diversified into the web, affiliate programs, torrent and P2P, illegal software, router DNS tampering, word processor vulnerabilities, spam mail, and storage media. In addition, the evasion technology according to the advancement of the production technology is also improved and the malicious code is produced more easily than before through the automatic generation program. In the past, the propagation speed of malicious codes was slow, the infection route was limited, and the propagation technology had a simple structure, so there was enough time to study countermeasures. However, current malicious codes have become very intelligent by absorbing technologies such as concealment and self-transformation, causing problems such as distributed denial of service attacks (DDoS), spam sending and personal information theft [1]. In general, the way to deal with malicious code is to analyze the malicious code and propose a solution

Manuscript received: October 24, 2021 / revised: November 2, 2021 / accepted: December 2, 2021
Corresponding Author: hwkim@syu.ac.kr
Tel:*** - **** - **** Fax: +82-2-3399-1791
Professor, Division of Computer Science & Engineering, Sahmyook Univ., Korea

based on the analyzed data. Therefore, in order to cope with malicious code, analysis of malicious code must be preceded [2]. One of important concerns in information security is to control information flow. It is whether to protect confidential information from being leaked, or to protect trusted information from being tainted [3]. A user is cheated to download android application from 3rd party app store as the installed mobile apps are updated after downloading malicious code from Command and Control (C&C) server activated by using attacker-oriented server-side polymorphic malware. Because the mobile apps downloaded by each user who undergoes this process are installed with slightly different code, those mobile apps can't be detected though conventional mobile anti-virus S/W [4, 5]. Unlike in the past, the types of malware damage are also diversifying due to system destruction through DDoS attacks, in addition to simple information leakage. Therefore, it is very important to prevent damage by detecting such malicious codes in advance. However, the signature detection technique, which is an existing malware detection technique, cannot respond when encountering malicious code with a changed attack pattern or new malicious code. In addition, it is difficult to perform static analysis on malicious code to which code obfuscation, encryption, and packing techniques are applied to make malicious code analysis difficult.

Therefore, in this paper, it is showed a method to detect malicious code through dynamic analysis and static analysis using Trojan-type Downloader/Dropper malicious code. Through this, it is showed a method to suggest to malicious code detection and countermeasures.

2. RELATED WORK

2.1 Dynamic Analysis

Analyze the behavior of the malicious code itself. In order to detect changes in the operating system, in the analysis environment, API functions are hooked in user mode and kernel mode, and when a specific event occurs, the system automatically calls A method of monitoring an event notification routine or the like is used.

Based on the relevant information, all executable files are executed in the order in which they are logged. And it measures how similar the behavior is to the execution type of the malicious code. If the executable file is diagnosed as malicious code, the system is restored in the reverse order of execution based on log values. There are emulators and sandboxes based on dynamic analysis. Virtualization technology is a technology that creates a virtual execution space on security equipment or security software and executes all suspicious files in this space. It protects the actual user environment by running the suspicious file on a virtualization basis before actually running it on the user's PC. The advantage of dynamic analysis is that it operates indirectly on virtual hardware, so it does not affect the real system.

As a disadvantage, it consumes a lot of system resources. A sandbox is also called an application emulator. Unlike an emulator, it runs a program on a real computer system. When the unknown program is executed, the CWMonitor.dll file is injected and all procedures of the executable file are traced. By hooking all Windows APIs, it determines whether it is a malicious code based on the collected information and return values [6, 7].

2.2 Static Analysis

Signature detection technology, which is a static analysis, analyzes the characteristics of already collected malicious code and generates a signature to detect the malicious code. Methods such as judging known malware by AV (Anti-Virus) scan or analyzing character strings in file headers and binaries are used. More professionally, it is determined whether there is a malicious code by a method such as an API call relationship analysis through a debugger. AhnLab developed 'Smart Defense' technology that manages a large-scale file

DB on a central server and collected more than 600 million normal and malicious file DBs. In addition, more than 50 billion malicious code features are extracted from this file DB and patterned to create a malicious code 'DNA map', and new and variant malicious codes are diagnosed through this 'DNA map'. The disadvantage of static analysis is that it is difficult to defend against new exploits. To overcome this, many packers are commercially available to bypass the static analysis base [8].

3. MALICIOUS CODE ANALYSIS

Malicious code was analyzed using Trojan-type Downloader/Dropper malicious code, and through the malicious code execution flow chart configured through IDA, three stages of initial analysis, dynamic analysis, and static analysis were performed. The tools used were PEiD, UPX, BinText, Ollydbg, IDA pro for dynamic analysis, Process Explorer and Wireshark for static analysis were used.

3.1 Analysis Target

The malware is a Trojan-type Downloader/Dropper malware, and most of the vaccines have updated detection patterns. The detailed information about the malicious code is as Table 1.

Table 1. Detailed information of malicious code

File Name	m1012.exe
Source File Name	m1012.exe
Diagnosis	Dropper/win32 OnlineGameHack.R39769 (AhnLab)
File Size	22,528 Byte
SHA256	B0245bb10d53f4c30256d3b6c916666b8dc22e1f7a1357f67490f5c6b4e6b28b

3.2 Malicious Code Execution Flowchart

The malware was analyzed as an online game hack as a Trojan-type Downloader/Dropper malware. Currently, most vaccines are updated and can be treated, and the progress of the malicious code is as follows.

1) When the first distributed malicious code file is executed, a duplicate file (xxxxxx.tmp) is created according to the reserved command, and the original file is deleted and terminated.

2) After that, the cloned file (xxxxxx.tmp) creates a specific process, terminates the antivirus process, creates a file similar to that of the existing .dll file (usp10.dll), and attempts to connect to a specific URL address.

3) When the connection is successful, the path of the temp file is created. Afterwards, it attempts to transmit to the specific URL address by hooking the keyboard input through the Keylogger function in the changed .dll file through the information values obtained from the URL. At this time, it is expected that another command will be executed from the accessed URL. Figure.1 is a flow chart of malicious code constructed through IDA.

3.3 Initial Analysis

In the initial analysis, the information of the malicious code was checked and whether the file was packed before detailed analysis was checked. The malicious code is an executable file in the form of an .exe and scanned through Virus Total to check whether the file has already been diagnosed. As a result of checking through Virus Total, it was confirmed that it was already diagnosed as a Trojan type of malicious code. Figure 2 shows the result of malicious code checked through Virus total.

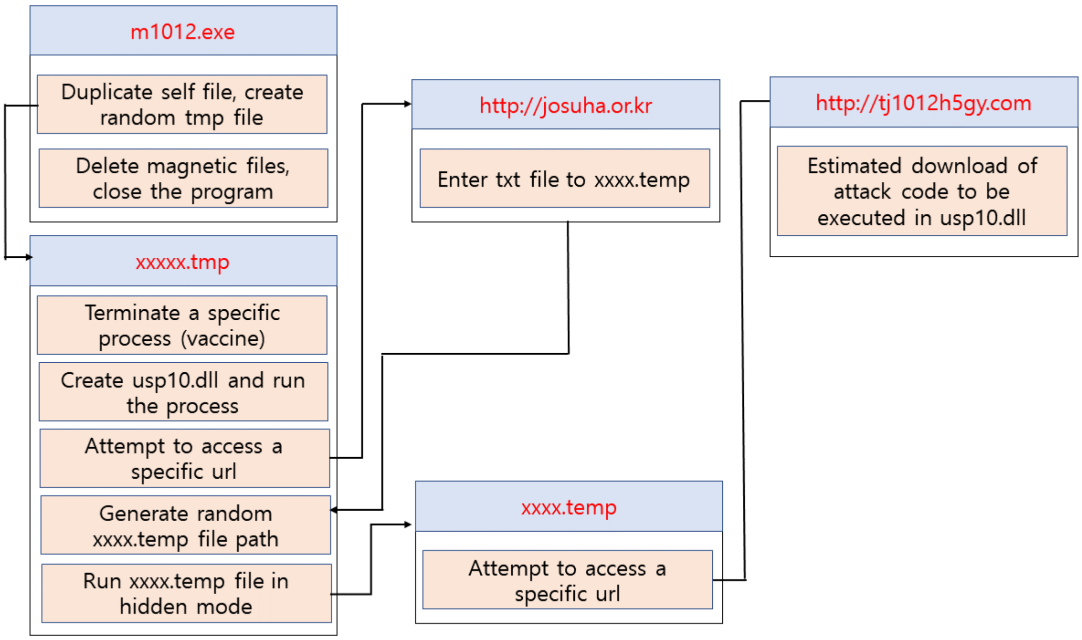

Figure 1. Execution Flowchart of malicious code

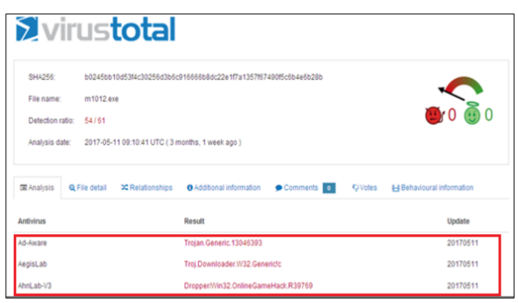

Figure 2. Malicious code through Virus total

3.4 Dynamic Analysis

In the dynamic analysis, it was checked whether there was a change in the process and network traffic when m1012.exe was executed. It was confirmed that the 02de080.tmp file was registered in the process list like Figure 3 when the program was executed.

After a few seconds, the program was run again to confirm that the .tmp file was terminated in the process. When you see that the name is different from the initially created .tmp file (02de080.tmp), you can guess that the .tmp file is created with a random file name by m1012.exe, and when you check the process operation of the .tmp file It was confirmed that the registry value was modified and the usp10.dll file was created in the C:\ Windows path. As a random .tmp process was registered, registry modifications and file creation were confirmed as Figure 4.

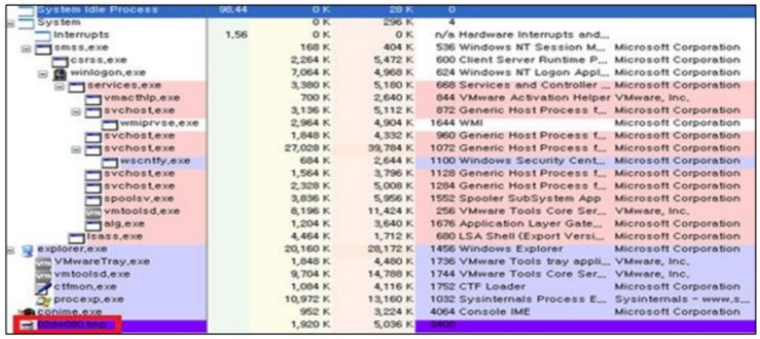

Figure 3. Register in process list

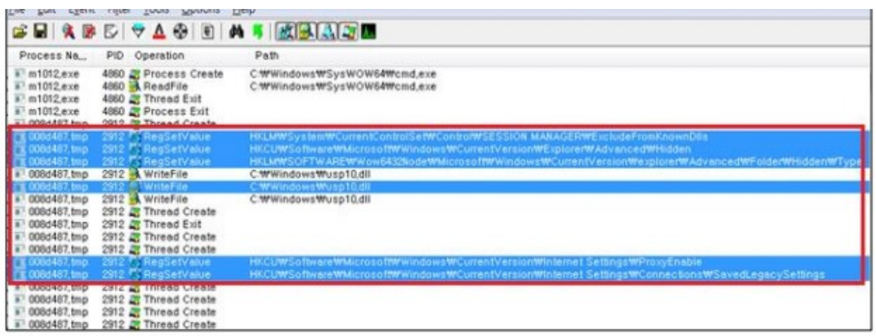

Figure 4. Registry modifications and file creation

In addition, as a random .tmp process was registered, changes in network traffic were confirmed as Figure 5.

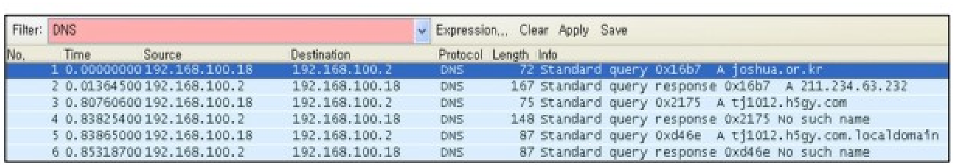

Figure 5. changes in network traffic

Network traffic trying to access two sites, Joshua.or.kr and tj1012.h5gy.com, was detected, and the IP address could not be found in DNS, so the request failed.

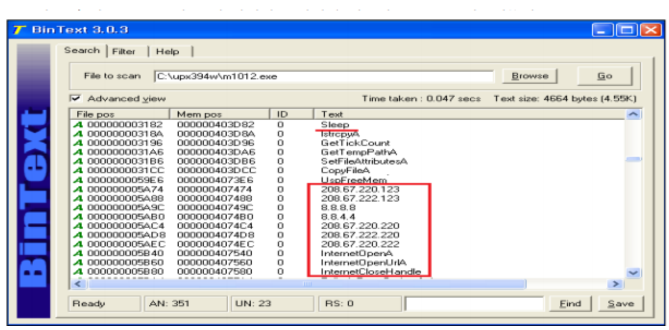

Figure 6. Suspected values of malicious purposes

3.5 Static Analysis

A String value is output to predict the execution of the program before reversing. Values suspected of malicious purposes such as forced termination and access to a specific URL were identified as Figure 6.

As a result of reversing, it was confirmed that a random file was created as Figure 7.

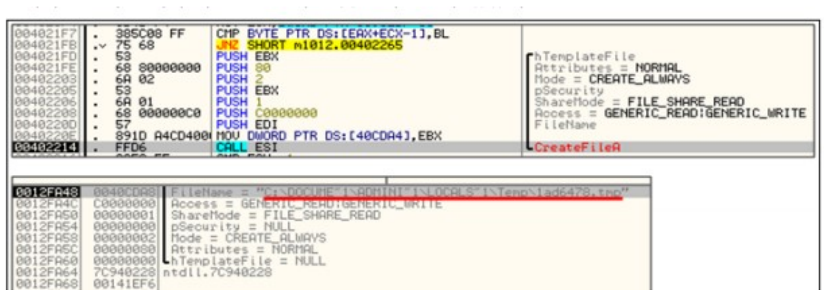

Figure 7. Creation of a random file

Afterwards, the m1012.exe file is deleted by itself, and the analysis of the m1012.exe file ends with self-deletion. Afterwards, the generated random .tmp files were analyzed. When executing a file, it was confirmed that a specific process was searched and terminated. Considering the process name, it can be inferred that the anti-virus process is terminated. After that, the usp10.dll file was created in the C:\Windows path. It also tries to connect to a specific URL (http://joshua.or.kr/data/m1012.txt). When the connection is successful, a .temp file with a random file name is created in the designated path. Data received from a specific URL (http://joshua.or.kr/data/m1012.txt) is recorded in the created .temp file. After running the .temp file in hidden mode, it was confirmed that an attempt was made to communicate with a specific URL as Figure 8.

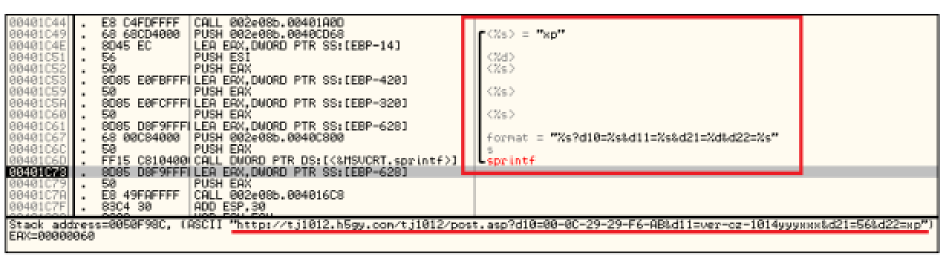

Figure 8. Attempt of communication with a specific URL

4. COUNTERMEASURES AND CONCLUSION

Suspected malicious code, m1012.exe creates a duplicate file, creates a process, terminates the antivirus process, and continues to execute malicious commands along with data downloaded through the induced URL connection. Therefore, it has the characteristics of a dropper and a downloader, and it is impossible to access the two URLs according to the analysis standard, and it is expected that there will be no additional damage as a vaccine patch for malicious code is made.

As a countermeasure, the malicious code already has a vaccine patch, so when the code is executed, the vaccine program itself deletes the executable file. Also, when a malicious code executable file is executed when the vaccine program is not running, the vaccine program that scans at a scheduled time can detect it. If it is not the case, there is a way to delete all the usp10.dll, arbitrary .tmp, and .temp files in the path, find the changed registry value, and manually restore it to before the execution of the malicious code.

The prevention method is that the malicious code does not work by itself unless the executable file is executed. It is important to improve the level of individual security awareness through education, because only security education can sufficiently prevent suspicious file downloads and unauthorized files from being installed and executed indiscriminately.

REFERENCES

[1] J. Y. Kim, J. L. Lee, E. J. Park, E. Y. Jang, and H.J. Kim (2009), “A study of Modeling and Simulation for Analyzing DDoS Attack Damage Scale and Defense Mechanism Expense,” Journal of the Korea Society for Simulation, 18(4), pp.39-47

[2] S. J Lee, K. H. Kim, Y. G. Shin, and J. H. Yi (2018), “Design and Implementation of Anti-reversing Code Evasion Framework for Intelligent Malware Analysis, Proceedings of the Korea Information Processing Society Conference, 25(2), pp.218-221. https://doi.org/10.3745/PKIPS.y2018m10a.218

[3] S. Ch Lee, S. G. Lee, H. Y. Oh, and S. M. Han (2019), “Piosk: A Practical Kiosk to Prevent Information Leakage,” International Journal of Advanced Smart Convergence, 8(2), pp.77-87. http://dx.doi.org/10.7236/IJASC.2019.8.2.77

[4] H. S. Lee, and H. W. Lee (2017), “Simulated Dynamic C&C Server Based Activated Evidence Aggregation of Evasive Server-Side Polymorphic Mobile Malware on Android,” International Journal of Advanced Smart Convergence, 6(1), pp. 1-8. https://doi.org/10.7236/IJASC.2017.6.1.1 †

[5] Vaibhav Rasgtogi, Yan Chen and Xuxian Jiang (2014), “Catch Me if You Can: Evaluating Android Anti-malware against Transformation Attacks,” IEEE Transactions on Information Forensics and Security, 9(1), pp. 99-108

[6] http://acc.daetoo.com/secu_info_view.asp?list=/secu_info_list.asp&seq=9533&pageno=18&v_num=714

[7] https://jianna6.tistory.com/entry

[8] http://blog.zeltser.com/post/4339793582/custom-signatures-for-malware-scan